(19) Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 10 2006 008 248 A1** 2007.08.23

(12) # Offenlegungsschrift

(21) Aktenzeichen: **10 2006 008 248.6**
(22) Anmeldetag: **22.02.2006**
(43) Offenlegungstag: **23.08.2007**

(51) Int Cl.[8]: ***G06F 9/46*** (2006.01)
G06F 12/14 (2006.01)
G06K 19/07 (2006.01)

(71) Anmelder:
Giesecke & Devrient GmbH, 81677 München, DE

(72) Erfinder:
Effing, Wolfgang, 85659 Forstern, DE; Spitz, Stephan, Dr., 82284 Grafrath, DE; Englbrecht, Erich, 85646 Anzing, DE; Hockauf, Robert, 81929 München, DE

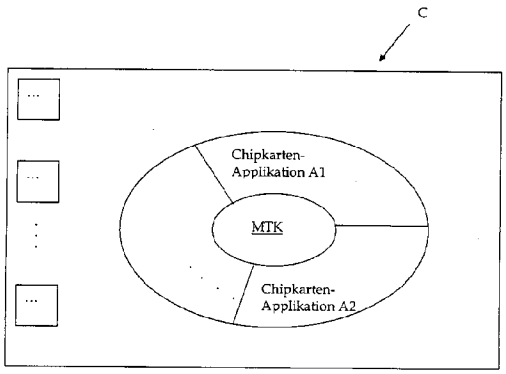
(56) Für die Beurteilung der Patentfähigkeit in Betracht zu ziehende Druckschriften:
Press Release-TASKING SLE88 tool Suite Offers Top-Notch Performance and Security for Smart Card,Technology, Amersfoort, Sept. 25th 2000, <http://directinsight.co.uk/news/pr27.html> In:<web.archive.org> am 11.03.2001, [recherchiert am 11.09.06],;
OHEIMB,D.,et.al.:A formal Security Model of the infineon SLE 88 Smart Card Memory Management.In:Proceedings of ESORICS 2003.Online: <http://david.von-oheimb.de/cs/papers/SLE88_MM.htm [Converene version], (recherchiert am 11.09.06);;

Die folgenden Angaben sind den vom Anmelder eingereichten Unterlagen entnommen

Rechercheantrag gemäß § 43 Abs. 1 Satz 1 PatG ist gestellt.

(54) Bezeichnung: **Betriebssystem für eine Chipkarte mit einem Multi-Tasking Kernel**

(57) Zusammenfassung: Die Erfindung betrifft ein Verfahren zum Betreiben einer Chipkarte (C), einen Mikroprozessor zum Einsetzen in die Chipkarte (C) und ein Computerprogrammprodukt sowie ein Verfahren zum Herstellen und/oder zum Warten einer Chipkarte (C), die mittels eines vorstehend beschriebenen Verfahrens betrieben wird. Dabei ist zentraler Multi-Tasking Kernel (MTK) vorgesehen, der den gesamten Betrieb der Chipkarte (C) steuert, so dass auf der Chipkarte (C) mehrere Applikationen (A) gleichzeitig aktiviert werden können, wobei eine Applikation (A) auch sicherheitstechnische Funktionen für die Chipkarte (C) realisieren kann.



Beschreibung

[0001] Die Erfindung liegt auf dem Gebiet der Chipkartentechnik und betrifft insbesondere ein Verfahren und ein System zum Betreiben von mobilen Datenträgern.

[0002] Mobile Datenträger finden heute vielfältige Einsatzgebiete vor, unter anderem als Chipkarte, wie z. B. die EC-Karte, die Anwendung als Zutritts- bzw. Zugangskontrolle, Chipkarten im Gesundheitswesen, im Bereich der Mobilfunktechnik als SIM-Karte (Subscriber Identity Module). Die SIM-Karte ist eine scheckkartengroße Identifizierungskarte für Teilnehmer eines Mobilfunkdienstes und wird auch als "Smartcard" bezeichnet. Darüber hinaus gibt es noch eine Vielzahl von weiteren Anwendungsmöglichkeiten von Chipkarten, etwa im Bereich der Navigationstechnik, bei digitalen Diktier- oder Kamerasystemen etc.

[0003] Üblicherweise umfasst der mobile Datenträger, insbesondere die Chipkarte, folgende Hardware-Ressourcen: Einen Mikroprozessor bzw. eine CPU (Central Processing Unit) zur Datenverarbeitung, mehrere Datenspeicher unterschiedlicher Art, wie dem RAM (Random Access Memory), dem ROM (Read Only Memory) und dem EEPROM (Electrical Eraseable Read Only Memory) und Schnittstellen für einen Datenaustausch zwischen den diversen Bauteilen, insbesondere zwischen dem Mikroprozessor und den Datenspeichern und gegebenenfalls zu weiteren Modulen auf der Chipkarte, sowie zu weiteren externen Modulen, die außerhalb der Chipkarte vorgesehen sind und mit der Chipkarte in Datenaustausch stehen sollen. Dabei kann es sich z. B. um Lesegeräte oder um komplexere Back-Office-Systeme handeln.

[0004] Je nach Anwendungsgebiet ist es möglich, auf der Chipkarte eine oder mehrere Applikationen laufen zu lassen. In diesem Fall, wenn also mehrere Anwendungen auf einer Chipkarte abgewickelt werden sollen, gewinnt der Sicherheitsaspekt verstärkt Bedeutung. Denn es muss sichergestellt sein, dass ein unberechtigter Datenzugriff sicher unterbunden werden kann. Bei mehreren Applikationen auf einer Karte steigt das Risiko insofern, da die Anwendungen in datentechnischer Hinsicht sicher voneinander entkoppelt und abgeschottet sein müssen. So muss z. B. sichergestellt sein, dass kein unerlaubter Zugriff auf einen bestimmten Speicherbereich über eine andere, fremde Applikation abgewickelt werden kann.

[0005] Karten, auf denen mehrere Applikation abgewickelt werden können, erfordern somit einen erhöhten Sicherheitsbedarf und sind komplexer; sie erfordern umfassendere Mechanismen zum Betreiben der Karte.

[0006] Das Betreiben der Chipkarte an sich und die Abwicklung von darauf laufenden Programmen bzw. Applikationen fallen in den Aufgabenbereich des Betriebssystems. Damit ist das Betriebssystem sozusagen eine Schnittstelle zwischen der eigentlichen Anwendungs-Software und der zugrunde liegenden Hardware der Chipkarte. Üblicherweise basieren die heutigen befehlsgetriggerten Chipkarten-Betriebssysteme auf dem im Stand der Technik bekannten Standard ISO-7816. In diesem Standard ist es vorgesehen, dass alle Funktionen bzw. Befehle des Betriebssystems und der Applikationen von Kommandos getriggert werden, die über eine externe Schnittstelle empfangen werden. Dabei werden die Befehle nur sequenziell, das heißt also nacheinander, ausgeführt. Mit anderen Worten gibt es damit nur einen Kontrollfluss für Prozesse in den jeweiligen Programmen. Derzeitige Implementierungen auf Chipkarten bestehen somit nur aus Prozessen mit einem einzigen Ausführungspfad bzw. mit einem einzigen Thread. Betriebssysteme, die ein Multi-Threading, also mehrere Ausführungspfade, unterstützen, sind für Chipkarten bisher nicht bekannt. Dies ist jedoch ein schwerwiegender Nachteil, der die Flexibilität beim Einsatz und beim Betreiben von Chipkarten deutlich einschränkt.

[0007] Um dem Nachteil der geringen Flexibilität entgegenzuwirken, ist es im Stand der Technik bei Chipkarten-Betriebssystemen der neueren Generation vorgesehen, dass der Programmcode zu beliebigen Zeitpunkten nachgeladen werden kann. Damit wird es möglich, auch nach Ausgabe der Karte einzelne Module bzw. Komponenten der Chipkarte gegen andere auszutauschen. Entsprechende Verfahren um Applikationen über eine ISO-7816-konforme Schnittstelle zu laden, sind z. B. in dem Standard "Global Platform Standard, Global Platform Card Specification V2.1.1" beschrieben.

[0008] Betriebssysteme, die das Nachladen von Programmcode ermöglichen, können grundsätzlich in zwei Kategorien unterteilt werden:

1. Betriebssysteme, bei denen es vorgesehen ist, einen von einem Compiler bereits übersetzten compilierten Code in die entsprechenden Dateien der Chipkarte zu laden. Dieser Ansatz birgt jedoch ein großes Sicherheitsrisiko, da es grundsätzlich möglich ist, dass ein nachgeladener Programmcode bei Mikrocontrollern, die ohne eine Memory-Management-Unit (kurz MMU) arbeiten, auch auf fremde Speicherbereiche von anderen Anwendungen zugreifen kann.
2. Betriebssysteme, die darauf basieren, dass nachzuladender Programmcode auf der Chipkarte interpretiert wird. Der Interpreter prüft dann während der Programmausführung, welche Speicherbereiche angesprochen werden und kann damit sicherstellen, dass keine unerlaubten Zugriffe auf Fremdanwendungen ausgeführt wer-

den. Zu den bekanntesten Lösungen dieses Ansatzes zählen die Javacard-Spezifikation (Javacard-Standard, Java Virtual Machine, Javasoft, JCS) und der C-Interpreter MEL (MEL steht für Multos Executeable Langauge) von Multos. Der grundsätzliche Nachteil dieses zweiten Ansatzes ist darin zu sehen, dass Interpreter jedoch grundsätzlich langsam arbeiten und dies zu einer schlechten Performance führen kann.

[0009] Heute auf dem Markt befindliche Mikrocontroller für Chipkarten sind in der Regel mit Prozessoren ausgestattet, die keine Speicherschutzmechanismen oder sonstige Überwachungsmöglichkeiten gegen unerlaubte Zugriffe haben. Um diesem Sicherheitsrisiko zu begegnen, wird der nachgeladene Programmcode nicht direkt ausgeführt, sondern nur indirekt, z. B. über eine so genannte virtuelle Maschine, die einzelne Programmbefehle (also den Bytecode) wiederum in plattformabhängigen Programmcode interpretiert, so dass Adressbereiche einzelner Programme bzw. Applikationen über die virtuelle Maschine oder über den Interpreter getrennt sind. In der virtuellen Maschine wird definiert, welche Zugriffe erlaubt sind, bzw. auf welche Daten eine Applikation Zugriff hat. Ein wesentlicher Nachteil ist jedoch darin zu sehen, dass grundsätzlich nur Interpretercode nachgeladen werden kann. Plattformabhängiger Programmcode, z. B. Treiber für Input-/Output-Schnittstellen können nach Kartenausgabe nicht mehr geladen werden.

[0010] Ein weiterer Nachteil ist darin zu sehen, dass die Sicherheit des Speicherschutzes auf der Sicherheit der virtuellen Maschine bzw. auf dem Interpreter beruht. Es ist zwar möglich, einen so genannten Bytecode-Verifier zur Verfügung zu stellen, der den Bytecode entsprechend überprüft. Nachteilig ist jedoch, dass die erforderlichen Überprüfungen des Verifiers aus Ressourcen- und/oder Performance-Gründen vorwiegend außerhalb der Chipkarte durchgeführt werden. Werden die Überprüfungen des Bytecode-Verifiers jedoch außerhalb der Chipkarte ausgeführt, ist dieser angreifbar.

[0011] Darüber hinaus liegt ein weiterer, in der Praxis nicht zu vernachlässigender Nachteil der bekannten Lösung in dem geringeren Umfang des Speicherschutzes. Der Speicherschutz bei bisherigen Chipkarten-Betriebssystemen beruht bei diesem Ansatz ausschließlich auf dem Interpreter bzw. auf der virtuellen Maschine. Ein Speicherschutz für umfassendere, komplexere Systeme, die z. B. aus mehreren virtuellen Maschinen bestehen, sind nicht bekannt. Mit anderen Worten gibt es deshalb keine Lösungen für Chipkarten-Betriebssysteme mit einem Speicherschutz beim Datenaustausch zwischen mehreren Interpretern bzw. mehreren virtuellen Maschinen auf einer Chipkarte.

[0012] Die vorliegende Erfindung hat sich deshalb zur Aufgabe gestellt, einen Weg aufzuzeigen, mit dem ein deutlich verbesserter Speicherschutz für Chipkarten-Betriebssysteme erreicht werden kann und der einen flexibleren Einsatz von Chipkarten ermöglicht. Darüber hinaus soll ein multi-taskingfähiges Betriebssystem für Chipkarten bzw. eine entsprechende Chipkarte und ein entsprechender Mikroprozessor bereitgestellt werden.

[0013] Diese Aufgabe wird mit einem Verfahren zum Betreiben eines mobilen Datenträgers, mit einem mobilen Datenträger, einem Mikroprozessor, einem Computerprogrammprodukt und mit einem Verfahren zum Herstellen und zum Warten eines mobilen Datenträgers gemäß den beiliegenden unabhängigen Patentansprüchen gelöst.

[0014] Die Aufgabe wird insbesondere durch ein Verfahren zum Betreiben eines mobilen Datenträgers gelöst, der mit folgenden Ressourcen ausgestattet ist:

- zumindest einem Mikroprozessor,
- zumindest einem Datenspeicher, der üblicherweise aus mehreren unterschiedlichen Datenspeicherbereichen besteht und
- Schnittstellen für einen Datenaustausch zwischen Mikroprozessor und Datenspeicher und/oder weiteren Modulen, die dem mobilen Datenträger zugeordnet sind, wobei auf dem mobilen Datenträger unterschiedliche Applikationen ausgeführt werden können, indem der mobile Datenträger eine zentrale Steuerungseinheit umfasst, die den Betrieb des mobilen Datenträgers, insbesondere die Ausführung der Applikationen, derart steuert und/oder überwacht, dass gleichzeitig mehrere Applikationen aktiv sein können, indem nach einem konfigurierbaren Scheduling-Mechanismus jeweils einer Applikation Ressourcen zugewiesen oder entzogen werden und/oder der Datenaustausch gesteuert wird.

[0015] Bei dem mobilen Datenträger handelt es sich üblicherweise um eine Chipkarte oder eine SIM-Karte oder um sonstige Mikroprozessorkarten, die in ein Endgerät, wie z. B. in ein mobiles Endgerät, wie ein Handy, eingesetzt werden können.

[0016] Die Erfindung kann auf unterschiedlichen Gebieten zum Einsatz kommen. So kann der erfindungsgemäße mobile Datenträger z. B. in Navigationssystemen, PDAs, digitalen Diktiersystemen, digitalen Kameras oder Telefonapparaten eingesetzt werden. Die hauptsächliche Ausführungsform der Erfindung betrifft jedoch eine Chipkarte und insofern ist der Begriff Chipkarte als hauptsächliches Ausführungsbeispiel für einen mobilen Datenträger zu verstehen.

[0017] Eine Chipkarte umfasst in der Regel folgen-

de Hardware-Ressourcen: Einen Mikroprozessor zur Datenverarbeitung, Datenspeicher und Schnittstellen. Es ist jedoch in alternativen Ausführungsformen ebenfalls möglich, hier weitere Ressourcen vorzusehen, wie z. B. einen mathematischen Coprozessor.

[0018] Bei den Schnittstellen handelt es sich in der Regel um Input-/Output-Schnittstellen. Auch hier können weitere Schnittstellen, etwa zu anderen externen und/oder internen Modulen, die dem mobilen Datenträger zugeordnet sind, vorgesehen sein.

[0019] Kernpunkt des erfindungsgemäßen Verfahrens zum Betreiben der Chipkarte ist die zentrale Steuerungseinheit, die in der bevorzugten Ausführungsform als Multi-Tasking Kernel ausgebildet ist und ein Bestandteil eines – bestehenden oder neuen – Betriebssystems für die Chipkarte ist. Der zentrale Multi-Tasking Kernel steuert und/oder kontrolliert Abläufe auf der Chipkarte und stellt geschützte Bereiche für die Ausführung zur Verfügung.

[0020] Im Gegensatz zu bekannten Betriebssystemen aus dem Stand der Technik, bei denen die Befehle des Betriebssystems bzw. der Applikationen von Kommandos, die über die externe Schnittstelle empfangen werden, getriggert und sequentiell nacheinander ausgeführt werden, werden gemäß der Erfindung alle Befehle durch den Multi-Tasking Kernel gesteuert. Der Multi-Tasking Kernel steuert den Betrieb der Chipkarte und die Abwicklung der auf ihr laufenden Prozesse derart, dass gleichzeitig mehrere Applikationen auf ein und derselben Chipkarte ausgeführt werden können. Dies wird erreicht, indem der Multi-Tasking Kernel nach einem Scheduling-Mechanismus, der vorzugsweise konfigurierbar ist, arbeitet. Der Scheduling-Mechanismus ist erlaubt – im Hinblick auf die Gesamtheit aller auf dem mobilen Datenträger aktivierbaren oder aktivierten Applikationen – eine optimierte Ausführung bzw. ein optimierter Betrieb des Datenträgers.

[0021] Der Multi-Tasking Kernel ermöglicht eine quasi-parallele Ausführung von mehreren auf der Chipkarte ablauffähigen software-basierten Applikationen. Er synchronisiert mittels des Scheduling-Mechanismus den Zugriff auf gemeinsame Betriebsmittel. Des Weiteren stellt er Mechanismen zum Zugriffsschutz bereit, die vor einem unberechtigten Zugriff auf Daten schützen und die dem Schutz gegen Beeinträchtigungen des Ablaufs dienen. Dies wird erreicht, indem der Multi-Tasking Kernel den Applikationen nach dem konfigurierbaren Scheduling-Mechanismus entsprechende Kontingente an Rechenzeit und an Ressourcen zuweist. Die Abwicklung bzw. Ausführung von Befehlen wird erfindungsgemäß also ausschließlich von dem zentralen Multi-Tasking Kernel getriggert.

[0022] Der Multi-Tasking Kernel bietet also die Möglichkeit, verschiedene Anwenderprogramme oder verschiedene Applikationen quasi gleichzeitig auszuführen, insbesondere mit der Option, Ressourcen (wie z. B. bestimmte Speicherbereiche im RAM oder im nicht-flüchtigen Speicher, Schnittstellen bzw. Input-/Output-Kanäle, kryptologische Module etc.) exklusiv einer Applikation zuzuweisen und sie bei Bedarf wieder zu entziehen. Dadurch kann eine Applikation im Zusammenspiel mit einem Chipkarten-Terminal z. B. eine "klassische" Chipkarten-Legacy-Aufgabe ausführen (z. B. Credit-/Debit-Befehle), während eine andere Anwendung im Hintergrund ausgeführt wird. Durch den Einsatz des Multi-Tasking Kernels wird eine – einseitige oder gegenseitige – Beeinflussung von aktiven Anwendungen sicher unterbunden, was vorteilhafter Weise die Sicherheit des Gesamtsystems erhöht.

[0023] Jeder Service bzw. jede Applikation verfügt über einen geschützten Adressraum. Es ist auch möglich, dass mehrere Applikationen bezüglich der Speicherverwaltung zusammengefasst werden, so dass sie in einem gemeinsamen Adressraum integriert werden. Vorteilhafter Weise kann erfindungsgemäß ein gesicherter Datenaustausch zwischen allen beteiligten Modulen der Chipkarte ermöglicht werden. Insbesondere ist der Datenaustausch zwischen den einzelnen, unterschiedlichen Applikationen vollständig durch den Multi-Tasking Kernel abgesichert, sowie ebenso der Datenaustausch mit anderen Modulen, die möglicherweise zu entsprechenden Schnittstellen an die Chipkarte angeschlossen werden, was insgesamt die Sicherheit des Gesamtsystems deutlich erhöht.

[0024] Erfindungsgemäß ist die Funktionalität der jeweiligen Applikationen bzw. Services nicht begrenzt. Services, die in einem geschützten Adressraum liegen, können sogar die komplette Funktionalität eines bisher gängigen Chipkarten-Betriebssystems (z. B. EC-Karte, Zutrittskontrolle, SIM-Karte, Gesundheitskarte etc.) in einer Umgebung nachbilden, die vor anderen Services geschützt ist. Der erfindungsgemäße Schutzmechanismus kann die Applikationen vollständig voneinander kapseln, so dass mehrere virtuelle Chipkarten auf einer Hardware-Plattform sicher koexistieren können.

[0025] Mit anderen Worten ist es durch die erfindungsgemäße Lösung mittels des Multi-Tasking Kernels möglich, in strikt voneinander getrennten Bereichen auf einer Hardware-Plattform, insbesondere auf ein und derselben Chipkarte, mehrere "virtuelle" Chipkarten anzubieten. Die einzelnen Applikationen, die jeweils "virtuelle Chipkarten" realisieren, sind dabei nicht mehr – wie bei klassischen Betriebssystemen aus dem Stand der Technik – um die Kommando-Schnittstelle gruppiert, sondern werden als Services über die Funktionen des zentralen Multi-Tasking Kernels gesteuert.

[0026] Ein weiterer zentraler Aspekt der vorliegenden Erfindung ist in dem Speicherschutz zu sehen. Erfindungsgemäß ist im Multi-Tasking Kernel ein Speicherschutz für plattformabhängigen Programmcode realisiert. Damit können die vorstehend erwähnten Nachteile des interpreter-basierten Speicherschutzes von den aus dem Stand der Technik bekannten Betriebssystemen überwunden werden.

[0027] In einer vorteilhaften Weiterbildung der Erfindung greift der Multi-Tasking Kernel auf einen Mechanismus zur Unterstützung der Trennung der Adressräume zu, insbesondere auf eine Memory-Management-Unit (kurz: MMU) und/oder auf eine Memory-Protection-Unit (kurz: MPU). Ein Vorteil dieses Mechanismus ist es, dass eine deutlich verbesserte Sicherheitssituation erreicht werden kann, im Vergleich zu einem rein software-basierten Interpreter oder einer virtuellen Maschine aus dem Stand der Technik.

[0028] Durch den Einsatz des Multi-Tasking Kernels an zentraler Stelle, das heißt auf der hierarchisch höchsten Prioritätsstufe, können mehrere, gleichzeitig aktive Applikationen auf einer Chipkarte ausgeführt werden. Dadurch wird die Möglichkeit eröffnet, dass einzelne Applikationen parallel und damit gleichzeitig auf nicht-konfligierende Ressourcen zugreifen können, und z.B. Daten über möglicherweise unterschiedliche Input-/Outpunt-Interfaces mit externen oder internen Systemen austauschen können. Kumulativ oder alternativ können auch Daten im Hintergrund von einer Applikation bearbeitet, insbesondere vorbereitet, werden, ohne dass dies explizit über eine externe Kommunikation getriggert wurde.

[0029] Der Multi-Tasking Kernel sieht vor, dass Prioritäten, insbesondere in Bezug auf einzelne Applikationen oder Applikationsgruppen, vergeben werden können und, dass eine Rechenzeitkontrolle erfolgt. Durch die Überwachung der Prioritäten und der Rechenzeit kann der Multi-Tasking Kernel sicherstellen, dass die einer Applikation zur Verfügung stehende Rechenzeit bzw. Ausführungszeit beschränkt ist und dass die durch den Multi-Tasking Kernel vorgegebenen Beschränkungen auch nicht manipuliert werden. Eine Beschränkung der Rechenzeit wird dadurch erreicht, dass der Verbrauch der Rechenzeit vom Multi-Tasking Kernel kontrolliert wird und die Rechenzeit in Form von Zeitquanten dezidiert den Applikationen zugewiesen wird. Die Manipulationssicherheit wird dadurch erreicht, dass ausschließlich der Multi-Tasking Kernel in einem höher privilegierten Betriebsmodus läuft, während alle Applikationen in einem hierarchisch niedriger angeordneten Anwendermodus arbeiten.

[0030] Neben der Synchronisation von aktiven Prozessen hat der Multi-Tasking Kernel jedoch noch weitere Aufgaben. So dient er erfindungsgemäß ebenso zur Verwaltung der Ressourcen der Chipkarte (wie z. B. Speicher und Schnittstellen). Die Ressourcen können von der Applikation beim ersten Laden oder dynamisch zur Laufzeit beim Multi-Tasking Kernel angefordert werden. Der Multi-Tasking Kernel entscheidet alleine und in erster Instanz, ob die Ressourcen exklusiv einer Applikation zugewiesen werden oder nicht. In nächster Instanz kann die Applikation weiteren Sub-Applikationen Rechte weiterreichen, die kleiner oder gleich der Rechte sind, die ihr zuvor vom Multi-Tasking Kernel eingeräumt worden sind. Somit ist auch eine Untervergabe bzw. einer Weitervergabe von Rechten an untergeordnete Sub-Applikationen vorgesehen.

[0031] Des Weiteren dient der Multi-Tasking Kernel dazu, Mechanismen zum sicheren Datenaustausch zwischen den einzelnen Applikationen bereitzustellen. Der durch den Multi-Tasking Kernel gesteuerte und/oder überwachte Datenaustausch zwischen den Applikationen basiert grundsätzlich auf dem Prinzip, dass der Datenaustausch ausschließlich unter Kontrolle des Multi-Tasking Kernels erfolgt. Hierfür sind grundsätzlich zwei Alternativen vorgesehen:

1. Die beteiligten Applikationen stehen in Datenaustausch bzw. können entsprechende Nachrichten über spezielle Multi-Tasking Kernel-Funktionsaufrufe austauschen.
2. Die beteiligten Applikationen können Daten über vordefinierte Speicherbereiche austauschen, die mehreren – in diesem Fall den aktiven – Applikationen gemeinsam zur Verfügung stehen.

[0032] Grundsätzlich ist es vorgesehen, dass jede Applikation selbst entscheidet, ob und welche Daten sie anderen Applikationen zur Verfügung stellt. Mit der erfindungsgemäßen Lösung wird also der Vorteil erreicht, dass unterschiedliche Anwendungen auf einer Chipkarte integriert werden können, die jedoch sicher voneinander abgeschottet sind.

[0033] Ein wesentlicher Vorteil der erfindungsgemäßen Lösung ist des weiteren darin zu sehen, dass der grundsätzliche Vorteil der Flexibilität, der unter anderem auch im Stand der Technik durch den Ansatz von nachladbarem Programmcode erreicht werden kann, auch mit der erfindungsgemäßen Lösung aufrechterhalten und sogar deutlich verbessert werden kann. Grundsätzlich ist es möglich, auch nach Kartenausgabe Komponenten, insbesondere Systemkomponenten, im Chipkarten-Betriebssystem auszutauschen oder neue Komponenten hinzuzufügen, wie z. B. Updates, oder Komponenten die dem Bug-fixing (der Fehlerbeseitigung) oder dergleichen dienen.

[0034] In der bevorzugten Ausführungsform der Erfindung ist es vorgesehen, dass grundsätzlich hardwarenahe Systemkomponenten – wie vorstehend erwähnt – in dem Chipkarten-Betriebssystem, die nicht über eine interpreter-basierte Programmiersprache

implementiert sind, wie z. B. Krypto-Routinen, Treiber für Input-/Output-Schnittstellen etc., nach Kartenausgabe ausgetauscht werden können. Dieser Austausch erfolgt, ohne dass ungewollte und/oder schädigende Einflüsse auf andere Komponenten ausgeführt werden, da der Speicherschutz des Multi-Tasking Kernels eine Beeinflussung von anderen Komponenten bzw. Betriebssystemkomponenten durch den ausgetauschten Service unterbindet. In einer vorteilhaften Weiterbildung der Erfindung ist es jedoch möglich, diesen Ansatz nicht nur auf Betriebssystemkomponenten anzuwenden, sondern auch auf andere Komponenten des Chipkartensystems, die somit auch nach Kartenausgabe ausgetauscht werden können, wenn es die jeweilige Anwendung ohne eine Verursachung von weiteren Fehlern ermöglicht. Damit kann das auf dem mobilen Datenträger basierende System sehr flexibel eingesetzt werden und ist leicht veränderbar.

[0035] Ein weiterer Vorteil der erfindungsgemäßen Lösung ist darin zu sehen, dass die Möglichkeiten des Datentransfers in Bezug auf die mobilen Datenträger erweitert werden können. Durch die Steuerung des Multi-Tasking Kernels wird es möglich, notwendige Kommunikationsprozesse optimiert zu triggern, dass eine parallele oder gleichzeitige Kommunikation mit internen oder externen Modulen über mehrere gleiche oder andersartige Hardware-Schnittstellen erfolgt. Mit anderen Worten kann ein auf dem erfindungsgemäßen Multi-Tasking Kernel basierendes Chipkartensystem die quasiparallele Ausführung von Programmcode dazu nutzen, Daten gleichzeitig über unterschiedliche Input-/Output-Schnittstellen, z. B. über eine kontaktlose Schnittstelle nach dem Standard ISO14443 oder nach dem NFC-Standard (Near Field Communication) und parallel dazu über eine kontaktbehaftete Schnittstelle nach dem ISO7816-Standard auszutauschen. Damit können insgesamt die Hardware-Ressourcen des mobilen Datenträgers deutlich besser ausgenutzt werden, was insgesamt zu erhöhter Abarbeitungsgeschwindigkeit des Datenträgers führt.

[0036] In der Regel sind zwei Betriebsmodi für den Betrieb des mobilen Datenträgers vorgesehen: Ein privilegierter Modus, in dem der zentrale Multi-Tasking Kernel abläuft, dem weitergehende Rechte eingeräumt sind als einem zweiten Modus, in dem grundsätzlich alle Anwendungen und/oder Prozesse bzw. Applikationen arbeiten. Je nach Anwendung des mobilen Datenträgers ist es jedoch ebenso möglich, hier noch weitere Privilegierungsstufen vorzusehen. Notwendig ist es jedoch, dass der zentrale Multi-Tasking Kernel jeweils am höchsten privilegiert ist, um eine zentrale Steuerung des gesamten Betriebs des Datenträgers zu ermöglichen.

[0037] Grundsätzlich basiert der erfindungsgemäße Multi-Tasking Kernel auf einem Scheduling-Mechanismus, der darauf ausgerichtet ist, im Hinblick auf die Gesamtheit aller auf dem Datenträger ablaufenden Prozesse (umfassend Betriebssystemprozesse und Applikationsprozesse) eine optimierte Ausführung oder Abwicklung aller Prozesse zu bewerkstelligen.

[0038] In einer bevorzugten Weiterbildung der Erfindung ist es vorgesehen, dass der Scheduling-Mechanismus auf einen Optimierungs-Algorithmus zugreift, der den Betrieb des Datenträgers hinsichtlich einem oder mehreren der folgenden Optimierungskriterien optimiert:

- eine Optimierung hinsichtlich Zeit, insbesondere bezüglich einer Abarbeitungsgeschwindigkeit, bezüglich einer Verweilzeit von Prozessen im Arbeitsspeicher und/oder einer Antwortzeit der Prozesse;
- eine Optimierung hinsichtlich der System-Ressourcen, insbesondere Hardware-Ressourcen;
- eine Optimierung hinsichtlich Speicherplatzbedarf und
- eine Optimierung hinsichtlich des notwendigen Datentransferns.

[0039] In alternativen Ausführungsformen sind noch weitere Optimierungskriterien konfigurierbar. Dies hat den Vorteil, dass die erfindungsgemäße Lösung sehr flexibel hinsichtlich der grundsätzlichen Prozessabwicklung ist. Das Betriebssystem der Chipkarte ist somit nicht auf ein bestimmtes Optimierungskriterium hin beschränkt. Üblicherweise wird der konfigurierbare Mechanismus aufgrund vordefinierter Eingabeparameter festgelegt. Die Eingabeparameter können über entsprechende Schnittstellen eingelesen werden. Alternativ ist es möglich, dass für bestimmte Anwendungen eine bevorzugte Behandlung der jeweiligen Applikation stattfindet. Dann kann der Multi-Tasking Kernel einer bestimmten Applikation alle oder ausgewählte Ressourcen exklusiv zuweisen. Die Ausbildung dieses Merkmals ist erfindungsgemäß jedoch nicht notwendig und lediglich optional.

[0040] In alternativen Ausführungsformen ist es auch möglich, andere Algorithmen zum Scheduling der Prozesse zum Betrieb des Datenträgers vorzusehen. So ist es z. B. möglich, das Scheduling-Verfahren durchsatz-basiert und/oder auslastungs-basiert auszubilden.

[0041] Damit die Aufgabe des Scheduling-Verfahrens umgesetzt werden kann ist es notwendig, dass der Multi-Tasking Kernel automatisch für jeden Prozess, dessen Ausführungszeit erfasst und kontrolliert. Des Weiteren wird eine Beschränkung für die Ausführungszeit jedes Prozesses vorgegeben (dies erfolgt nach dem Mechanismus: "Wie lange darf welcher Prozess dauern?"). Daraufhin ist es möglich, dass der Scheduling-Mechanismus automatisch die Ausführungszeit für eine jeweilige Applikation be-

schränkt, indem der Verbrauch der Rechenzeit kontrolliert und indem die Einhaltung der Beschränkungen überwacht wird. Optional kann auch eine verschachtelte bzw. verschränkte Verarbeitung von Prozessen gefahren werden, so dass insgesamt die Ausführungszeit aller notwendigen Prozesse auf dem Datenträger optimiert werden kann. Nach dem optimierten Scheduling-Verfahren wird dann Rechenzeit an den jeweiligen Prozess bzw. an die jeweilige Applikation zugewiesen.

[0042] Im Rahmen des Schedulings ist es möglich, dass einzelnen Applikationen und/oder einzelnen Prozessen Prioritäten zugewiesen werden, die bei dem Scheduling berücksichtigt werden. Darüber hinaus ist es möglich, dass ein Prozess seine Priorität an untergeordnete Subprozesse weitervererbt.

[0043] In diesem Zusammenhang ist auf einen weiteren wesentlichen Aspekt der erfindungsgemäßen Lösung hinsichtlich eines verbesserten Sicherheitsansatzes hinzuweisen. Wie bereits erwähnt, können Chipkarten auch in Endgeräten, wie z. B. in Mobiltelefonen eingesetzt werden und sind in diesem Fall als SIM-Karte ausgebildet. In diesem Anwendungsfall sind üblicherweise noch weitere Schnittstellen, wie z. B. USB- oder MMC-Schnittstellen an dem SIM-Kontakten im Mobiltelefon vorgesehen, über die weitere Sicherheits-Devices angesprochen werden können, z. B. SecureMMC-Karten etc. Werden Chipkarten in Mobiltelefonen oder anderen mobilen Endgeräten eingesetzt, so ist es also häufig der Fall, dass Sicherheitsmodule bzw. Sicherheitskomponenten, die Sicherheitsüberprüfungen durchführen sollen, in dem System verteilt ausgebildet sind. Diese Verteilung von sicherheitskritischen Funktionen auf unterschiedliche Systeme und Komponenten in den chipkartenbezogenen Bauteilen bzw. Geräten führt zu mehreren Nachteilen. So erhöhen sich zum Einen die Herstellungskosten, da mehrere Hardware-Elemente zum Einsatz kommen müssen und zum Anderen ist die Fehleranfälligkeit des Systems insgesamt deutlich erhöht, da durch die Vielzahl der Module eine erhöhte Anfälligkeit für Sicherheitslücken besteht. Darüber hinaus ist es durch die bisherige verteilte Realisierung von sicherheitsrelevanten Funktionen notwendig, in einem erhöhten Maß Daten zu transferieren. Dies führt wiederum zu einer Sicherheitslücke, da grundsätzlich jeder Datentransfer ein Sicherheitsrisiko in sich birgt. Mit dem multi-tasking-fähigen erfindungsgemäßen Betriebssystem wird es jedoch möglich, dass die Chipkarte, die mit diesem System betrieben wird, mehr Funktionen übernehmen kann, unter anderem neben den klassischen standardisierten Funktionen (im vorstehenden Beispiel neben der reinen SIM-Funktionalität) noch weitere sicherheitstechnische Funktionen. Zum anderen wird es mit diesem Ansatz möglich, alle Sicherheitsfunktionen zentral an einer Stelle des Systems zu integrieren.

[0044] In einer bevorzugten Ausführungsform der Erfindung ist es deshalb vorgesehen, ein Sicherheitsmodul vorzusehen, das so genannte Trust-Management-Modul (im Folgenden kurz als TMM abgekürzt). Dieses Modul wird ebenfalls von dem Multi-Tasking Kernel gesteuert. Das TMM-Modul kann unterschiedliche sicherheitskritische Aufgaben in einer geschützten Umgebung übernehmen, wie z. B. neben der reinen SIM-Funktionalität eine DRM-Authentisierung (DRM steht für Digital Rights Management und betrifft ein Kontrollsystem zur Überprüfung einer Übertragung von geschützten bzw. zu schützenden Inhalten). Darüber hinaus können andere Autorisierungs-Mechanismen unterstützt werden.

[0045] Das TMM-Modul kann sowohl physikalisch als Hardwarebauteil ausgebildet sein. Es ist jedoch auch möglich, das Modul oder einzelne Funktionalitäten des Moduls als Software bzw. als Computerprogrammprodukt vorzusehen, die auf einem bestimmten Sicherheitsprozessor z. B. auf einem Secure-ARM-Core laufen.

[0046] An dieser Stelle soll explizit darauf hingewiesen werden, dass alle von dem zentralen Multi-Tasking Kernel ansprechbaren Module sowohl in Hardware als auch in Software realisiert werden können, was die Flexibilität des Systems insgesamt erhöht.

[0047] Ein wichtiger Vorteil im Zusammenhang mit den Sicherheitsaspekten des TMM-Moduls ist darin zu sehen, dass Sicherheitsfunktionen flexibel nachgeladen werden können. Darüber hinaus ist es möglich, die Funktionalität, die das erfindungsgemäße TMM-Modul unterstützt, im Unterschied zum Stand der Technik deutlich zu erhöhen. Damit kann das erfindungsgemäß vom Multi-Tasking Kernel betriebene TMM-Modul deutlich mehr Funktionalitäten bieten, als es z.B. von Javacard-Applets bekannt ist. Dazu gehören plattformabhängige Treiber für Sicherheitsprotokolle, wie IPSec oder SSL/TLS oder Authorisierungssysteme für das Digital Rights Management im Zusammenhang mit Multimedia-Inhalten.

[0048] Ein wesentlicher, vorteilhafter Aspekt des erfindungsgemäßen TMM-Moduls ist des Weiteren darin zu sehen, dass es auch selbst aktiv Sicherheitsüberprüfungen durchführen kann. Dies ist bei bisherigen TPM-Modulen nicht der Fall (Trusted Platform Module, kurz TPM, ist ein Sicherheitsstandard, der von der Trusted Computing Group entwickelt worden ist; die Module dieses Standards werden grundsätzlich als System-on-Chip realisiert). Im Gegensatz zu den bekannten TPM-Modulen aus dem Stand der Technik wird das erfindungsgemäße TMM-Modul nicht als reiner Slave betrieben, der nur auf Anfragen einer anderen Instanz antwortet, sondern das TMM-Modul kann auch selbständig Aktionen steuern. Dieses Merkmal der selbstständigen Steuerung ist jedoch nicht zwingend und nur fakultativ.

[0049] Insgesamt kann durch den erfindungsgemäßen Betrieb der Chipkarte mit einem TMM-Modul ein verbesserter Speicherschutz erzielt werden. Durch das multi-tasking-fähige Betriebssystem können unterschiedliche sicherheitskritische Aufgaben in einem Sicherheitssystem, insbesondere in einem spezifischen Chipkartenprozessor, untergebracht und damit realisiert werden.

[0050] Es gibt unterschiedliche Ausführungsformen, in denen das TMM-Modul auf dem mobilen Datenträger realisiert sein kann. So ist es möglich, das TMM-Modul steckbar oder fest verdrahtet zu realisieren, oder es kann bereits auf dem Halbleiter integriert sein. Darüber hinaus ist es möglich, das Modul über unterschiedliche Protokolle, wie z. B. über das ISO7816 T=0 oder T=1, über eine USB- oder über eine MMC-Schnittstelle oder allgemein über den Prozessorbus anzuschließen. Darüber hinaus ist es optional möglich, einen TCP-IP/Stack auf dem Schicht2-Protokoll vorzusehen.

[0051] Weitere Lösungen der eingangs erwähnten Aufgabe liegen in einem Betriebssystem oder in Betriebssystemkomponenten, in einem mobilen Datenträger, in einem Mikroprozessor zum Einsetzen in den mobilen Datenträger, in einem Computerprogrammprodukt und in einem Verfahren zum Herstellen oder zum Warten des mobilen Datenträgers gemäß den beiliegenden Hauptansprüchen. Grundsätzlich ist in diesem Zusammenhang darauf hinzuweisen, dass die Beschreibung der Erfindung auf eine Beschreibung des erfindungsgemäßen Verfahrens gestützt ist. Vorteilhafte Ausführungsform, Vorteile und Weiterbildungen, die im Zusammenhang mit dem Verfahren beschrieben werden, gelten entsprechend auch für die anderen Lösungen der Erfindung, insbesondere durch den mobilen Datenträger, den Mikroprozessor und das Computerprogrammprodukt. Demnach können die vorstehend genannten Lösungen auch mittels der Merkmale aus den Unteransprüchen zu dem erfindungsgemäßen Verfahren weitergebildet sein.

[0052] Die vorstehend beschriebenen, erfindungsgemäßen Ausführungsformen des Verfahren können auch als Computerprogrammprodukt ausgebildet sein, mit einem von einem Computer lesbaren Medium und mit einem Computerprogramm und zugehörigen Programmcode-Mitteln, wobei der Computer nach Laden des Computerprogramms zur Durchführung des oben beschriebenen, erfindungsgemäßen Verfahrens veranlasst wird.

[0053] Eine alternative Aufgabenlösung sieht ein Speichermedium vor, das zur Speicherung des vorstehend beschriebenen, computer-implementierten Verfahrens bestimmt ist und von einem Computer lesbar ist.

[0054] Eine weitere Lösung der Aufgabe ist darin zu sehen, dass das oben beschriebene Verfahren als Betriebssystem oder Betriebssystemkomponente für einen mobilen Datenträger ausgebildet ist, der gemäß zumindest einem Merkmal des Verfahrens betrieben wird.

[0055] Zusätzliche, vorteilhafte Ausführungsformen ergeben sich aus den Unteransprüchen.

[0056] In der folgenden detaillierten Figurenbeschreibung werden nicht einschränkend zu verstehende Ausführungsbeispiele mit deren Merkmalen und weiteren Vorteilen anhand der Zeichnung besprochen. In dieser zeigen:

[0057] Fig. 1 eine schematische, übersichtsartige Darstellung eines erfindungsgemäßen Multi-Tasking Kernels, der den Betrieb des mobilen Datenträgers gemäß einer Ausführungsform der Erfindung steuert,

[0058] Fig. 2 eine übersichtsartige Darstellung einer Aktivierung von Applikationen durch den erfindungsgemäßen Multi-Tasking Kernel gemäß einer bevorzugten Ausführungsform und

[0059] Fig. 3 eine übersichtsartige Darstellung einer möglichen Strukturierung von Bauteilen eines erfindungsgemäßen Datenträgers.

[0060] In der bevorzugten Ausführungsform und im Folgenden ist ein mobiler Datenträger als Chipkarte C ausgebildet. Die Anwendungen der Chipkarte C sind jedoch grundsätzlich nicht begrenzt und können auf dem Gebiet des Zahlungsverkehrs, des Finanzwesens, der Zutrittskontrolle liegen. Des Weiteren ist es möglich, dass die Chipkarte C zum Einsatz in weiteren Geräten, z. B. mobilen Endgeräten wie Telefonen, eingesetzt wird und es sich insbesondere um eine erfindungsgemäß erweiterte SIM-Karte handelt.

[0061] Grundsätzlich wird die Chipkarte C selbst und die auf ihr ablaufenden Applikationen A durch ein Betriebssystem gesteuert. Bei bisherigen Chipkarten-Betriebssystemen waren die Programm-Module des Betriebssystems üblicherweise in einem ROM-Speicherbaustein gespeichert (Read-Only-Memory ROM). Um den Nachteilen einer Speicherung von Betriebssystemkomponenten ausschließlich in den ROM-Speicher zu begegnen, kann es vorgesehen sein, einzelne Betriebssystemkomponenten auch in anderen Speicherbereichen, wie z.B. durch einen Arbeitsbereich in EEPROM, zu lösen. Zu den Hauptaufgaben eines Chipkarten-Betriebssystems zählen der Datenaustausch mit der Chipkarte, die Ablaufsteuerung der auszuführenden Befehle, die Dateiverwaltung und die Verwaltung und Ausführung von sicherheitstechnischen Funktionen und Algorithmen, wie kryptografischen Schlüsseln etc. Darüber hinaus ist es möglich, in einem Bereich der hierar-

chisch über den Applikationsbefehlen angeordnet ist, einen Interpreter oder ein Prüfprogramm für ausführbare Dateien vorzusehen. Der Interpreter dient dazu, die in diesen Dateien enthaltenen Programme auszuführen oder sie zu interpretieren. Mit dieser Art von Betriebssystemen ist es möglich, Programmcode auch zu einem späteren Zeitpunkt, insbesondere nach Kartenausgabe, nachzuladen.

[0062] Wie in Fig. 3 beispielhaft dargestellt, umfasst die Chipkarte C einen eingebetteten Mikrocontroller, der alle Aktivitäten der Chipkarte C triggert, steuert und überwacht. Die wichtigsten, typischen Bausteine eines Chipkarten-Mikrocontrollers sind der Mikroprozessor MP, alle Schnittstellen SS der Chipkarte C, insbesondere der Adress- und Datenbus und die Datenspeicher DS, die alle unterschiedlichen Speicherarten umfassen, wie RAM, ROM und EEPROM. Die Schnittstellen SS der Chipkarte C umfassen alle Input-/Output-Schnittstellen für die Chipkarte C und betreffen somit den gesamten Datentransfer, der in Bezug auf die Chipkarte C anfällt.

[0063] Erfindungsgemäß ist zusätzlich zu diesen Bauteilen eine zentrale Steuerungseinrichtung MTK vorgesehen, die insbesondere durch den Multi-Tasking Kernel gebildet wird. In Fig. 3 ist der Multi-Tasking Kernel MTK als separates Bauteil auf der Chipkarte C dargestellt. Dies soll verdeutlichen, dass der Multi-Tasking Kernel MTK – im Gegensatz zu den bekannten Chipkarten-Betriebssystemen – als zusätzliches Bauteil vorgesehen ist. In der Regel wird er jedoch nicht als separates, selbstständiges Bauteil vorgesehen sein, sondern in anderen Bereichen der Chipkarte als separates Modul integriert sein. Insbesondere wird er als modulare, separate Betriebssystemkomponente zusätzlich zu dem bisherigen Betriebssystem der Chipkarte C vorgesehen sein.

[0064] Je nach Anwendung der Chipkarte C umfasst diese mehrere Applikationen bzw. Services A, die auf der Chipkarte ablaufen sollen.

[0065] Im Rahmen dieser Erfindung sind die Begriffe "Applikation" A und "Service" A synonym zu verstehen. Eine Applikation A umfasst mehrere Befehle bzw. Prozesse, die zu unterschiedlichen Zeitpunkten ausgeführt werden müssen oder können. Eine Anwendung umfasst in der Regel mehrere Applikationen A. Es ist jedoch grundsätzlich auch möglich, dass eine sehr einfache Anwendung lediglich aus einer einzigen Applikation A besteht.

[0066] Durch den zentralen Multi-Tasking Kernel MTK wird die Möglichkeit geschaffen, auf ein und derselben Hardware-Plattform einer Chipkarte C mehrere, sozusagen "virtuelle" Chipkarten anzubieten. Die einzelnen virtuellen Chipkarten sind dabei streng voneinander getrennt, da alle Applikationen und Befehle über den zentralen Multi-Tasking Kernel MTK gesteuert werden. Eine einseitige oder wechselseitige Beeinflussung von aktiven Applikationen oder Anwendungen wird damit durch den Multi-Tasking Kernel MTK sicher unterbunden.

[0067] Der Multi-Tasking Kernel MTK weist den Applikationen A entsprechende Kontingente an Rechenzeit und Ressourcen nach einem konfigurierbaren Scheduling-Verfahren zu. Wie in Fig. 1 exemplarisch dargestellt, stehen alle Applikationen A bzw. Chipkartenservices A mit dem Multi-Tasking Kernel MTK in Datenaustausch und werden von diesen gesteuert und ausgeführt. In Fig. 1 ist angedeutet, dass das Scheduling des Multi-Tasking Kernels MTK zeitbasiert ist. Dies soll durch die zeitscheibenartige Darstellung in Fig. 1 verdeutlicht werden. Der Multi-Tasking Kernel MTK überwacht und steuert zur Ausführungszeit die Abwicklung der einzelnen Applikationen. Mittels des konfigurierbaren Scheduling-Mechanismus wird jeweils einer Applikation automatisch ein Kontingent an Rechenzeit und Ressourcen zur Verfügung gestellt, die von der jeweiligen Applikation A genützt werden können. Die Ausführungszeit jeder Applikation A ist somit automatisch in einem konfigurierbaren Maß beschränkt.

[0068] In der bevorzugten Ausführungsform der Erfindung muss der Multi-Tasking Kernel MTK eine Analyse des bestehenden Systemzustandes mit entsprechend zu triggernden Applikationen A ausführen und muss daraufhin die gesamte Abwicklung bzw. den Betrieb der Chipkarte C steuern, so dass im Hinblick auf die Gesamtheit aller auszuführenden Befehle eine optimierte Ausführung erfolgt. Dabei sind die Optimierungskriterien konfigurierbar: z. B. eine Optimierung hinsichtlich Zeit, Systemressourcen, Speicherplatz, Stromverbrauch etc.

[0069] Vor Ausführung einer jeweiligen Applikation A erfasst der Multi-Tasking Kernel MTK, wie viel Rechenzeit für die Ausführung notwendig ist und wie viel und/oder welche Ressourcen erforderlich sind. Sollen nun mehrere Applikationen A ausgeführt werden, so kann der Multi-Tasking Kernel MTK aufgrund der Analyse der Rechenzeit und der benötigten Ressourcen aller Applikationen einen optimierte Abwicklung einzelner Prozesse, die den jeweiligen Applikationen A zugeordnet sind, triggern. Hat z. B. eine erste Applikation A_1 die Aufgabe, Daten über eine kontaktlose Schnittstelle an ein externes Modul weiterzuleiten und hat z. B. eine zweite Applikation A_2 die Aufgabe, Daten von einem weiteren externen Modul über eine kontaktbehaftete Schnittstelle zu empfangen, so kann der Multi-Tasking Kernel MTK eine quasi-parallele, das heißt gleichzeitige Aktivierung der beiden Applikationen A_1 und A_2 veranlassen, da die beiden Applikationen auf unterschiedliche Ressourcen (in diesem Fall unterschiedliche Schnittstellen SS) zugreifen. Damit kann der bei bisherigen Systemen aus dem Stand der Technik sequenzielle Abarbeitungs-

pfad von Befehlen parallelisiert werden und auf mehrere gleichläufige Prozesse aufgeteilt werden, so dass insgesamt die Performance gesteigert werden kann.

[0070] Durch das Betreiben der Chipkarte C mit dem erfindungsgemäßen Multi-Tasking Kernel MTK ist es möglich, mehrere nebenläufige Threads zu realisieren, falls keine konkurrierenden bzw. konfligierenden Zugriffe auf gleiche Ressourcen notwendig sind. Es kann vorgesehen sein, dass der Multi-Tasking Kernel MTK auf ein zeitbasiertes Scheduling zugreift, falls er einen konkurrierenden Zugriff von unterschiedlichen Applikationen zur selben Zeit auf gleiche Ressourcen erfasst. Das zeitbasierte Scheduling sieht dann vor, dass die Gesamtheit der auszuführenden Prozesse der beiden Applikationen A_1 und A_2 so gesteuert wird, dass insgesamt (also im Hinblick auf die Gesamtheit der beiden Applikationen A_1 und A_2) eine optimierte, insbesondere zeit-optimierte, Ausführung ermöglicht wird. Damit ist es z. B. möglich, Daten einer Anwendung A_1 im Hintergrund vorbereiten zu lassen, während eine andere Anwendung A_2 z. B. mit einem externen System über Schnittstellen SS kommuniziert.

[0071] In Fig. 2 ist schematisch dargestellt, wie der Multi-Tasking Kernel MTK unterschiedliche Applikationen A_1, A_2, A_3 auf optimierte Weise aktiviert.

[0072] Die in Fig. 2 dargestellten Applikationen A_1 und A_2 werden jeweils durch externe Systeme verursacht. Dies kann z. B. eine Kontoumsatzanfrage im Rahmen einer finanziellen Anwendung sein. Kerngedanke der vorliegenden Erfindung ist es, dass die einzelnen Anfragen und auszuführenden Befehle nicht mehr direkt ausgeführt werden, sondern alle über den zentralen Multi-Tasking Kernel MTK gesteuert werden. Aufgrund des Scheduling-Algorithmus aktiviert der Multi-Tasking Kernel MTK einzelne Prozesse der Applikationen A_1, A_2 und A_3,..., A_i derart, dass eine optimierte Ausführung der Gesamtheit aller Applikationen A_i ermöglicht wird. In Fig. 2 ist dies dargestellt, indem die vom Multi-Tasking Kernel MTK aktivierten Applikationen mit einem dicken vertikal verlaufenden Strich gekennzeichnet sind, während die jeweiligen Prozesse bzw. Befehle einer Applikation A, die gerade nicht aktiv sind, bzw. vom Multi-Tasking Kernel MTK nicht aktiviert wurden, lediglich mit einem dünnen vertikalen Strich gekennzeichnet sind. So ist ersichtlich, dass der Multi-Tasking Kernel MTK auf Anfrage des externen Systems 1B als erstes die Applikation A_1 aktiviert und daraufhin einen Befehlszyklus der Applikation A_2, der von dem externen System 1A verursacht worden ist. Im Anschluss daran wird wieder zur Applikation A_1 zurückgekehrt, um daraufhin die Applikation A_3 zu beginnen und anschließend die Applikation A_2 zu beenden. Im Anschluss an das Beenden der Applikation A_2 werden die restlichen Befehle der Applikation A_3 ausgeführt. Insgesamt ist so ein zeitoptimiertes Scheduling der Gesamtheit der Applikationen A_i möglich.

[0073] Ein zentraler Aspekt der vorliegenden Erfindung liegt in verbesserten Sicherheitsvorkehrungen, insbesondere in einem verbesserten Speicherschutz. In diesem Fall ist es vorgesehen, dass in zumindest einer Applikation A alle sicherheitsrelevanten Befehle bzw. Prozesse, die im Rahmen des Betriebs der Chipkarte C notwendig sind, zusammengefasst und integriert werden. Diese Applikation A bzw. dieses Modul wird TMM-Modul (Trust Management Module) genannt. In diesem Modul sind also alle sicherheitsrelevanten Funktionen und Befehle zusammengefasst. Es ist möglich, weitere Sicherheitsfunktionen flexibel über bestimmte Protokolle nachzuladen.

[0074] Erfindungsgemäß kann der Inhalt des TMM-Moduls flexibel konfiguriert werden. Damit ist es möglich, je nach Anwendung, unterschiedliche Sicherheitsmechanismen zu aktivieren und/oder zu deaktivieren, um eine optimale sicherheitstechnische Abdeckung der Chipkarte C für den jeweiligen Anwendungsfall zu erzielen. Das TMM-Modul ist erfindungsgemäß so ausgelegt, dass es auch aktiv Sicherheitsüberprüfungen durchführen kann und somit nicht – wie im Stand der Technik – als reiner abhängiger Prozess betrieben wird.

[0075] Ein weiterer, wesentlicher Vorteil der erfindungsgemäßen Lösung ist darin zu sehen, dass die sicherheitstechnischen Prozesse, die im TMM-Modul integriert sind, optimiert in den Ablauf bzw. in den gesamten Betrieb der Chipkarte C eingefasst werden können. Dies hat den Hintergrund, dass gewisse sicherheitstechnische Überprüfungen nur zu einem bestimmten Zeitpunkt im Systemablauf Sinn machen. So ist z. B. eine Authentifizierungsmaßnahme nur vor Beginn einer Transaktion sinnvoll, während weitere sicherheitstechnische Maßnahmen auch zu einem späteren Zeitpunkt ausgeführt werden können. Die optimale, insbesondere zeitoptimierte, Steuerung aller Prozesse auf der Chipkarte C wird durch den Multi-Tasking Kernel MTK kontrolliert und überwacht.

[0076] Für die erfindungsgemäße Lösung ist es auch möglich, dass spezifische Sicherheitsmechanismen, die beispielsweise nur bei einer bestimmten Anwendung einzusetzen sind, durch das flexibel konfigurierbare TMM-Modul beim Betreiben der Chipkarte C eingesetzt werden können. Im Unterschied zu den Verfahren aus dem Stand der Technik war bisher keine Anpassung hinsichtlich sicherheitstechnischer Maßnahmen an eine bestimmte Art von Anwendung möglich. Dieser Nachteil wird mit der erfindungsgemäßen Lösung vollständig beseitigt.

[0077] Die erfindungsgemäße Lösung ist vorteilhafter Weise unabhängig von der jeweiligen Plattform der Chipkarte C und insbesondere unabhängig da-

von, ob eine virtuelle Maschine zum Einsatz kommt oder nicht oder ob die virtuelle Maschine off-card oder on-card realisiert ist.

[0078] An dieser Stelle sei noch einmal darauf hingewiesen, dass die vorstehende, detaillierte Figurenbeschreibung im Zusammenhang mit der erfindungsgemäßen Lösung durch das Verfahren beschrieben worden ist. Vorteilhafte Weiterbildungen, Alternativen, Vorteile und Merkmale, die im Zusammenhang mit dem Verfahren geschildert worden sind, sind ebenso auf die anderen Lösungen der Aufgabe zu lesen und somit insbesondere auf den mobilen Datenträger, den Mikroprozessor, das Computerprogrammprodukt und auf das Verfahren zum Herstellen und/oder zum Warten des mobilen Datenträgers anwendbar. Die vorstehend erwähnten Module, Komponenten und Einheiten des beschriebenen Verfahrens können sowohl in einer verkaufsfähigen Einheit bereits integriert sein, sie können jedoch auch als eigenständiges, separates Produkt nachträglich integriert werden, ohne dass weitere Maßnahmen an bestehenden Produkten notwendig sind.

[0079] Die in dieser detaillierten Figurenbeschreibung beschriebenen Ausführungsformen sollen lediglich Beispiele darstellen und können vom Fachmann auf verschiedenste Weise modifiziert werden, ohne dass der Bereich der Erfindung verlassen wird. Für einen einschlägigen Fachmann ist es insbesondere offensichtlich, dass die Erfindung auch als heterogenes System und teilweise oder vollständig in Software und/oder Hardware und auf mehrere physikalische Produkte – dabei insbesondere auf Computerprogrammprodukte – verteilt realisiert sein kann.

Patentansprüche

1. Verfahren zum Betreiben eines mobilen Datenträgers (C), der mit folgenden Ressourcen ausgestattet ist: einem Mikroprozessor (MP), einem Datenspeicher (DS), Schnittstellen (SS) für einen Datenaustausch zwischen Mikroprozessor (MP) und Datenspeicher (DS) und/oder weiteren Modulen, die dem mobilen Datenträger (C) zugeordnet sind, wobei auf dem mobilen Datenträger (C) unterschiedliche Applikationen (A) gleichzeitig ausgeführt werden können, indem der mobile Datenträger (C) mittels einer zentralen Steuerungseinheit (MTK) betrieben wird, die den Betrieb des mobilen Datenträgers (C) derart steuert und/oder überwacht, indem jeweils einer Applikation (A) nach einem Scheduling-Mechanismus Ressourcen zugewiesen und/oder der Datenaustausch gesteuert wird.

2. Verfahren nach Anspruch 1, dadurch gekennzeichnet, dass in der Steuerungseinheit (MTK) ein hardware-unterstützter Schutzmechanismus für den Datenspeicher (DS) ausgebildet ist.

3. Verfahren nach Anspruch 1 oder 2, dadurch gekennzeichnet, dass der Scheduling-Mechanismus konfigurierbar ist.

4. Verfahren nach zumindest einem der vorstehenden Patentansprüche, dadurch gekennzeichnet, dass der Betrieb des mobilen Datenträgers (C) mittels der Steuerungseinheit (MTK) derart gesteuert wird, dass der Datenaustausch zwischen unterschiedlichen Anwendungen und/oder zwischen unterschiedlichen Applikationen (A), ausschließlich über die Steuerungseinheit (MTK) gesteuert und/oder ausgeführt wird.

5. Verfahren nach zumindest einem der vorstehenden Patentansprüche, dadurch gekennzeichnet, dass der Betrieb des mobilen Datenträgers (C) mittels der Steuerungseinheit (MTK) derart gesteuert wird, dass eine parallele oder gleichzeitige Kommunikation mit externen Modulen über mehrere gleiche oder unterschiedliche Hardware-Schnittstellen (SS) erfolgt.

6. Verfahren nach zumindest einem der vorstehenden Patentansprüche, dadurch gekennzeichnet, dass der mobile Datenträger (C) mehrere Betriebsmodi aufweist und dem Betriebsmodus mit der die Steuerungseinheit betrieben wird wenigstens so viele Rechte eingeräumt werden wie den übrigen Betriebsmodi.

7. Verfahren nach zumindest einem der vorstehenden Patentansprüche, dadurch gekennzeichnet, dass der Scheduling-Mechanismus automatisch jeweils eine Ausführungszeit für eine Applikation (A) beschränkt, indem ein Verbrauch von Rechenzeit kontrolliert und Rechenzeit an die jeweiligen Applikationen (A) zugewiesen wird.

8. Verfahren nach zumindest einem der vorstehenden Patentansprüche, dadurch gekennzeichnet, dass der konfigurierbare Mechanismus Rechte berücksichtigt, die jeweils einer Applikation (A) vergeben werden können, und die an untergeordnete Sub-Applikationen weitergereicht werden können.

9. Verfahren nach zumindest einem der vorstehenden Patentansprüche, dadurch gekennzeichnet, dass jede Applikation (A) jeweils einzeln und/oder zusammen mit anderen Applikationen einen geschützten Adressraum des Datenspeichers (DS) hat.

10. Verfahren nach zumindest einem der vorstehenden Patentansprüche, dadurch gekennzeichnet, dass die zentrale Steuerungseinrichtung (MTK) zusätzlich alle sicherheitsrelevanten Funktionen steuert und/oder überwacht, die möglicherweise auf verschiedenen Bauteilen des mobilen Datenträgers (C) verteilt realisiert sind.

11. Mobiler Datenträger (C), auf dem unterschiedliche Applikationen (A) ausgeführt werden können, umfassend folgende Ressourcen:
– Mikroprozessor (MP),
– Datenspeicher (DS),
– Schnittstellen (SS) für einen Datenaustausch zwischen Mikroprozessor (MP) und Datenspeicher (DS) und/oder weiteren Modulen, die dem mobilen Datenträger (C) zugeordnet sind
dadurch gekennzeichnet, dass der mobile Datenträger (C) eine zentrale Steuerungseinheit (MTK) mit einem Scheduler umfasst, wobei die Steuerungseinheit (MTK) den Betrieb des mobilen Datenträgers (C), insbesondere die Ausführung der Applikationen (A), derart steuert und/oder überwacht, dass gleichzeitig mehrere Applikationen (A) aktiv sein können, indem der Scheduler nach einem konfigurierbaren Mechanismus jeweils einer Applikation Ressourcen zuweist und/oder den Datenaustausch steuert.

12. Mikroprozessor (MP) zum Einsetzen in einen mobilen Datenträger (C), auf dem unterschiedliche Applikationen (A) ausgeführt werden können, wobei der Mikroprozessor (MP) folgende Ressourcen umfasst:
– Datenspeicher (DS),
– Schnittstellen (SS) für einen Datenaustausch zwischen Mikroprozessor und Datenspeicher und/oder weiteren Modulen, wobei dem Mikroprozessor (MP) eine zentrale Steuerungseinheit (MTK) mit einem Scheduler zugeordnet ist, und wobei die Steuerungseinheit (MTK) den Betrieb des mobilen Datenträgers (C), insbesondere die Ausführung der Applikationen (A), derart steuert und/oder überwacht, dass gleichzeitig mehrere Applikationen (A) aktiv sein können, indem der Scheduler nach einem konfigurierbaren Mechanismus jeweils einer Applikation (A) Ressourcen zuweist und/oder den Datenaustausch steuert.

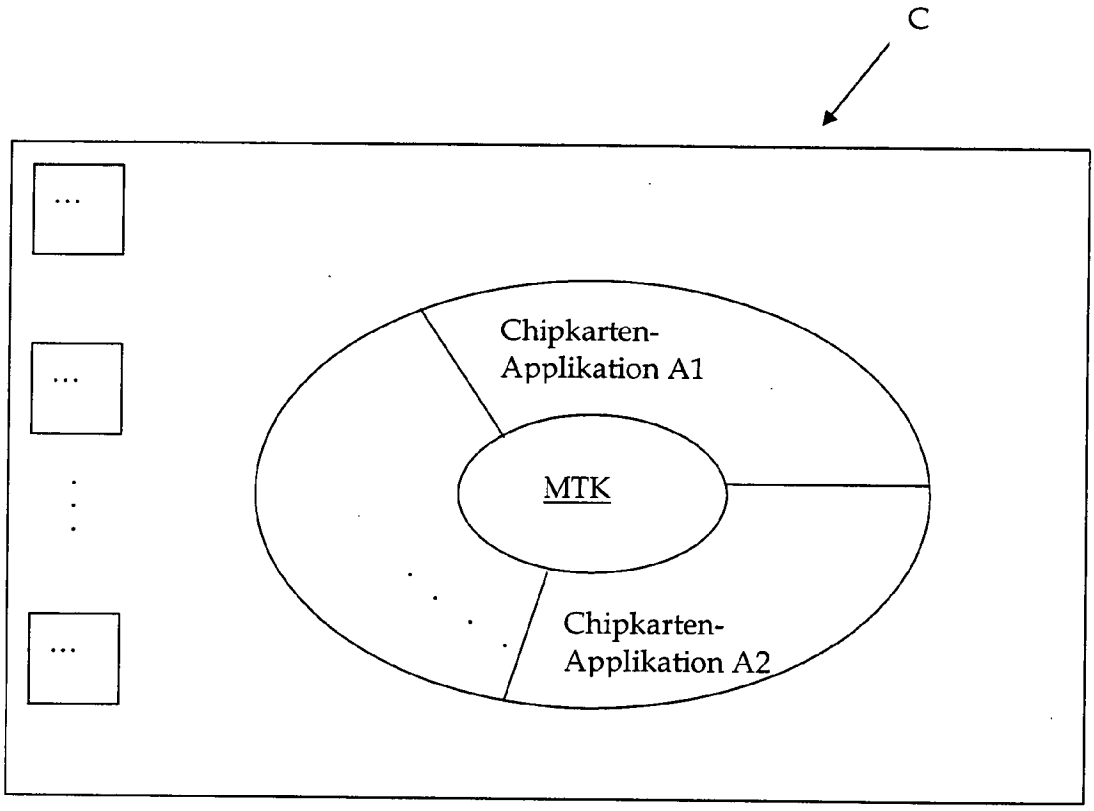
13. Computerprogrammprodukt, welches direkt in einen Datenspeicher (DS) eines programmierbaren mobilen Datenträgers (C) oder in eine dem mobilen Datenträger (C) zugeordnete Steuereinrichtung ladbar ist, mit Programmcodemitteln, um alle oder ausgewählte Schritte eines Verfahrens nach einem der vorstehenden Verfahrensansprüche auszuführen, wenn das Computerprogrammprodukt in dem mobilen Datenträger (C) oder in der Steuereinrichtung ausgeführt wird.

14. Computerprogrammprodukt nach Anspruch 12, dadurch gekennzeichnet, dass das Computerprogrammprodukt als Betriebssystem oder Betriebssystemkomponente ausgebildet ist.

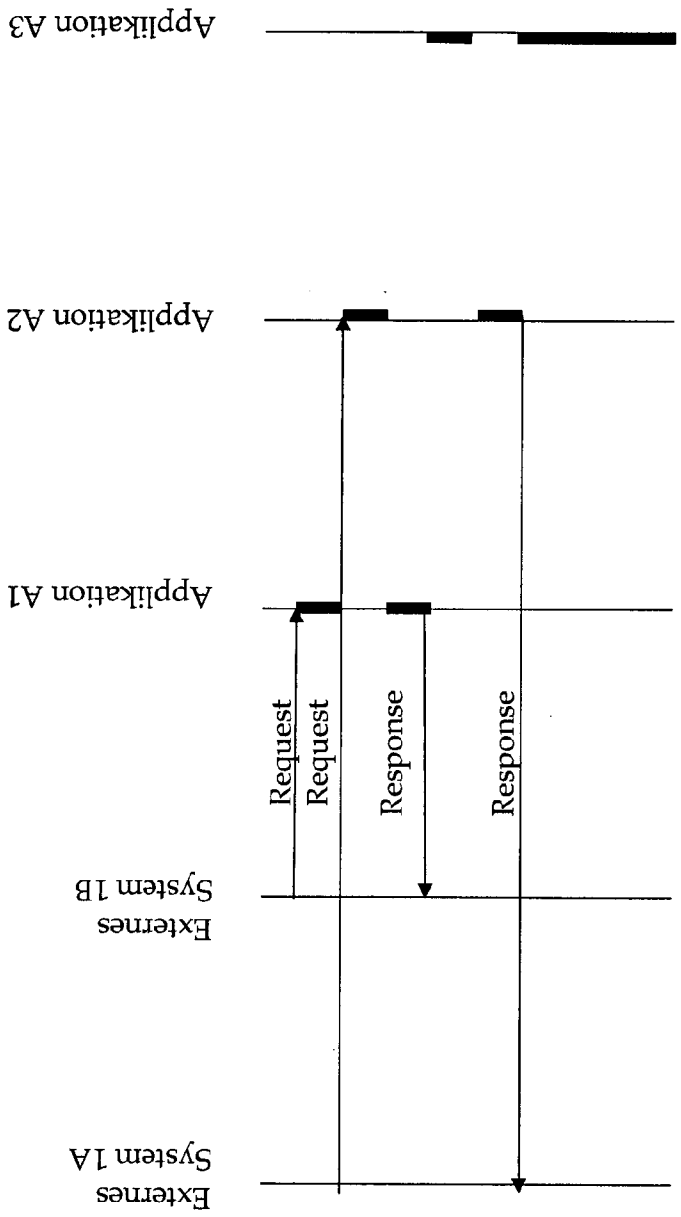
15. Verfahren zum Herstellen und/oder zum Warten eines mobilen Datenträgers (C), der mit einem Verfahren gemäß den Merkmalen der vorstehenden Verfahrensansprüche betrieben wird, wobei Komponenten, insbesondere software-basierte Betriebssystemkomponenten, auch nach Ausgabe des mobilen Datenträgers (C) durch andere Komponenten ersetzt werden können.

Es folgen 3 Blatt Zeichnungen

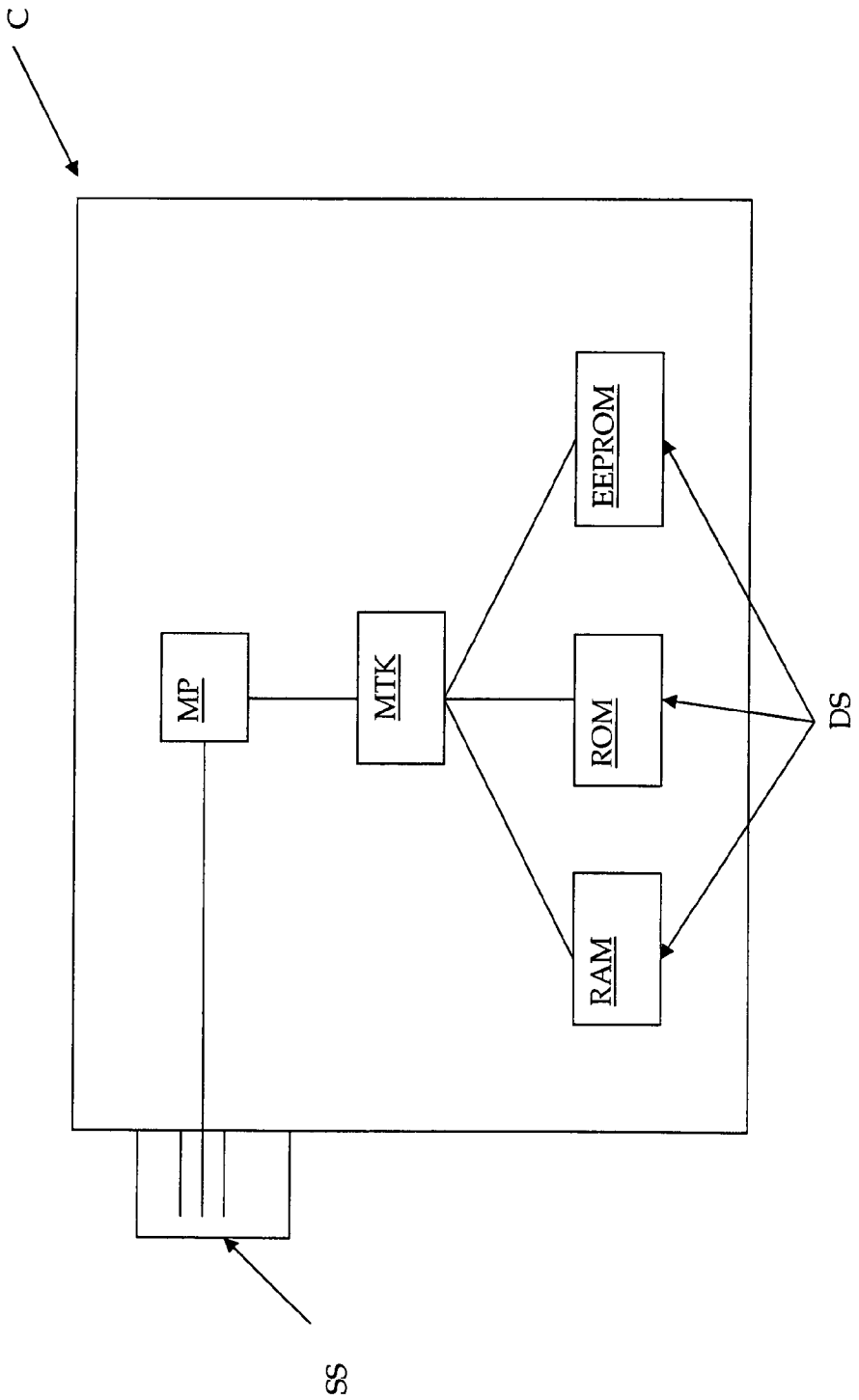
Anhängende Zeichnungen



Figur 1


Application A3
Application A2
Application A1
Request
Request
Response
Response
Externes System 1B
Externes System 1A

Figur 2



Figur 3

DERWENT-ACC-NO: 2007-720703

DERWENT-WEEK: 200768

TITLE: Mobile data carrier e.g. chip card, operating method, involves controlling and/or monitoring operation of mobile data carrier by central control unit such that application e.g. service, is allotted according to scheduling mechanism resource

INVENTOR: EFFING W; ENGLBRECHT E ; HOCKAUF R ; SPITZ S

PATENT-ASSIGNEE: GIESECKE & DEVRIENT GMBH[GIESN]

PRIORITY-DATA: 2006DE-10008248 (February 22, 2006)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE
DE 102006008248 A1	August 23, 2007	DE
WO 2007096153 A1	August 30, 2007	DE

DESIGNATED-STATES: AE AG AL AM AT AU AZ BA BB BG BR BW BY BZ CA CH CN CO CR CU CZ DE DK DM DZ EC EE EG ES FI GB GD GE GH GM GT HN HR HU ID IL IN IS JP KE KG KM KN KP KR KZ LA LC LK LR LS LT LU LV LY MA MD MG MK MN MW MX MY MZ NA N G NI NO NZ OM PG PH PL PT RO RS RU SC SD SE SG SK SL SM SV SY TJ TM TN TR TT TZ UA UG US UZ VC VN ZA ZM ZW AT BE BG BW CH CY CZ DE DK EA EE ES FI FR GB GH GM GR HU IE IS IT KE LS LT LU LV MC MW MZ NA NL OA PL PT RO SD SE SI SK SL SZ TR TZ UG ZM ZW

APPLICATION-DATA:

PUB-NO	APPL-DESCRIPTOR	APPL-NO	APPL-DATE
DE102006008248A1	N/A	2006DE-10008248	February 22, 2006
WO2007096153A1	N/A	2007WO-EP001511	February 21, 2007

INT-CL-CURRENT:

TYPE	IPC DATE
CIPP	G06F9/46 20060101
CIPP	G06F9/48 20060101
CIPS	G06F12/14 20060101
CIPS	G06K19/07 20060101

ABSTRACTED-PUB-NO: DE 102006008248 A1

BASIC-ABSTRACT:

NOVELTY - The method involves implementing different applications (A1, A2) e.g. services, on a mobile data carrier (C) e.g. chip card, at the same time, and operating the mobile data carrier by a central control unit (MTK). The operation of the mobile data carrier is controlled and/or monitored by the central control unit in such a manner that each application is allotted in accordance to a scheduling mechanism resource, and/or a data exchange between a microprocessor and a data storage is controlled.

DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

(1) a mobile data carrier comprising a microprocessor

(2) a computer program product loadable directly into a data storage of a programmable mobile data carrier or in a control device, comprising program code medium for implementing a method for operating a mobile data carrier

(3) a method for producing and/or for holding a mobile data carrier.

USE - Used for operating a mobile data carrier e.g. chip card, EC card, subscriber identity module (SIM) card and microprocessor card, that is utilized in a navigation system, personal digital assistant, digital dictation system, digital camera, telephone set and health care, where the microprocessor card is utilized in a mobile terminal e.g. mobile phone.

ADVANTAGE - The method provides the memory protection for a chip card-operating system in a clearly improved manner and enables flexible utilization of the chip card.

DESCRIPTION OF DRAWING(S) - The drawing shows a schematic representation of a multi-tasking kernel that controls the operation of a mobile data carrier. `(Drawing includes non-English language text)`

Applications (A1, A2)

Mobile data carrier (C)

Central control unit (MTK)

CHOSEN-DRAWING: Dwg.1/3

TITLE-TERMS: MOBILE DATA CARRY CHIP CARD OPERATE METHOD CONTROL MONITOR CENTRAL UNIT APPLY SERVICE ALLOT ACCORD SCHEDULE MECHANISM RESOURCE

DERWENT-CLASS: T01 T04 W01

EPI-CODES: T01-F02C2; T01-S03; T04-K03A; W01-C01D3C; W01-C01D3D;

SECONDARY-ACC-NO:

Non-CPI Secondary Accession Numbers: 2007-567842